TCP Header Format:

The Transmission Control Protocol (TCP) header is the first 24 bytes of a TCP segment that contains the parameters and state of an end-to-end TCP socket. The TCP header is used to track the state of communication between two TCP endpoints. Since TCP segments are inserted (encapsulated) in the payload of the IP packet the TCP header immediately follows the IP header during transmission. TCP does not need to keep track of which systems are communicating, it only needs to track which end to end sockets are currently open. Internet Protocol handles the logical addressing, routing and host-to-host connectivity.

TCP uses port numbers on each side of the connection to track the connection endpoints, state bits such as SYN, ACK, RST, FIN, sequence numbers and acknowledgement numbers to track the communication at each step in transmission.

An example of a TCP header is shown below.

TCP Packet Diagram

Source Port (0 - 65535)								Destination Port (0 - 65535)
Sequence Number (0 - 4294967295)								
Acknowledgement Number (0 - 4294967295)								
Data Offset	*Reserved*	URG	ACK	PSH	RST	SYN	FIN	Window
Checksum (CRC-Check)								Urgent Pointer
Options								Padding
Data								

TCP 'Packet' Field	Bits	Usage
Source Port	16	The TCP Source Port is the port number used by the computer sending the TCP segment and is usually a number above 1024 (but not always).
Destination Port	16	The TCP Destination Port is the port number used by the computer receiving the TCP packet and is usually a number below 1024 (but not always).
Sequence Number	32	Used for segmentation of application data into TCP segments and reassembling them on the other side. The sequence number helps the TCP software on both sides keep track of how much data has been transferred and to put the data back into the

		correct order if it is received in the wrong order, and to request data when it has been lost in transit.
Data Offset	4	The TCP Data Offset indicates number of bytes into the TCP packet where data can be found Thus, it actually indicates the number of bytes in the TCP header and allows the receiver to jump directly to the data.
Reserved	6	
URG Urgent Flag	1	
ACK Acknowledgement Flag	1	Used during 3-way handshake and data transfers.
PSH Push Flag	1	Used for TCP push, which returns the buffer to the user application. Used primarilly in streaming.
RST Reset Flag	1	Used to reset a TCP connection
SYN Synchronize Flag	1	Used during 3-way handshake
FIN End of data	1	Indicates end of the TCP session
Window	16	Number of octets in the TCP header
Checksum	16	A Cyclic Redundancy Check (CRC) checksum is calculated by the sender and added to this field before transmission. This field is used by the receiver to verify the integrity of the data in the TCP payload and rejects data that fails the CRC check.
Urgent Pointer	16	Points to the end of "urgent" data in the packet, but this field only exists if the URG flag is set.
Options	Varies	
Padding	Varies	
Data	Varies	This field contains a segment of data from the user application, such as part of an email or web page.

<u>Question TCP:</u>
The following is a dump of a TCP header in hexadecimal format.

05320017 00000001 00000000 500207FF 00000000

a. What is the source port number?
b. What is the destination port number?
c. What is the sequence number?
d. What is the acknowledgment number?
e. What is the length of header?
f. What is the type of segment?
g. What is the window size?

<u>Solution 2</u>
TCP header itself is of 10 fields as below and size may vary between 20 to 60bytes

1.Source port - 2 bytes
2.destination port - 2 bytes
3.SEQ NUM-4 bytes
4.ACK NUM- 4 bytes
5.HLEN-1 word
6.RESERVED-6bits
7.CONTROL-6bits
8.WINDOW SIZE-2 bytes
9.CHECKSUM-2 bytes
10.URGENT POINTERS-2bytes

here's example problem from book Forouzan

TCP header(in hex)=05320017 00000001 000000000 500207FF 00000000
since each hex = 4 bits , we need to first split the above hex as such
05 32 00 17 00 00 00 01 00 00 00 00 50 02 07 FF 00 00 00 00

source port is 2 bytes take 05 32 = 1330
next 2 bytes as destination address 00 17 == 23 (default TCP port)
next 4 bytes as sequence number 00 00 00 01 ==1
next 4 bytes as ack 00 00 00 00 == 0
next 4 bits as HLEN 5 =>5 -- this indicates number of sets of 4 bytes which makes the header lenght = 20bytes..
next 6 bits are reserved i.e.0 =0000and 2 bits from hex 0
next 6 bits are control bits = remaining 2 bits from hex 0 and 4 bits of 2
next 2 bytes indicate the window length 07 FF == 2047 bytes
Checksum 2 bytes 00 00 = 0
Urgent pointer 2bytes 00 00 =0

UDP Header

A computer may send UDP packets without first establishing a connection to the recipient. A UDP datagram is carried in a single IP packet and is hence limited to a maximum payload of 65,507 bytes for IPv4 and 65,527 bytes for IPv6. The transmission of large IP packets usually requires IP fragmentation. Fragmentation decreases communication reliability and efficiency and should therefore be avoided.

To transmit a UDP datagram, a computer completes the appropriate fields in the UDP header (PCI) and forwards the data together with the header for transmission by the IP network layer.

The UDP header consists of four fields each of 2 bytes in length

- **Source Port** (UDP packets from a client use this as a service access point (SAP) to indicate the session on the local client that originated the packet. UDP packets from a server carry the server SAP in this field)
- **Destination Port** (UDP packets from a client use this as a service access point (SAP) to indicate the service required from the remote server. UDP packets from a server carry the client SAP in this field)
- **UDP length** (The number of bytes comprising the combined UDP header information and payload data)
- **UDP Checksum** (A checksum to verify that the end to end data has not been corrupted by routers or bridges in the network or by the processing in an end system. The algorithm to compute the checksum is the Standard Internet Checksum algorithm. This allows the receiver to verify that it was the intended destination of the packet, because it covers the IP addresses, port numbers and protocol number, and it verifies that the packet is not truncated or padded, because it covers the size field. Therefore, this protects an application against receiving corrupted payload data in place of, or in addition to, the data that was sent. In the cases where this check is not required, the value of 0x0000 is placed in this field, in which case the data is not checked by the receiver.

Question UDP:

The following is a dump of a UDP header in hexadecimal form: **06 32 00 0D 00 1C E2 17**
What is the
(a) Source port number
(b) Destination port number
(c) Total length of the UDP
(d) Length of the data
(e) Considering that an IP frame can have a maximum total length of 65 535 bytes, what is the maximum length of the data in a UDP frame?
(f) What is the type of segment?

The UDP header has four parts, each of two bytes. That means we get the following interpretation of the header.
(a) Source port number = 063216 = 1586
(b) Destination port number = 000D16 = 13
(c) Total length = 001C16 = 28 bytes
(d) Since the header is 8 bytes the data length is 28 − 8 = 20 bytes.
(e) The IP header is minimum 20 bytes, which gives the maximum payload 65515 bytes. To fit a UDP frame in this with header of 8 bytes we get data 65515−8 = 65507 bytes.
(f) The *Daytime Protocol* is a simple protocol that allows clients to retrieve the current date and time from a remote server.